INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Universe Detectors for Sybil Defense in Ad Hoc Wireless Networks

Adnan Vora[†] — Mikhail Nesterenko[†] — Sébastien Tixeuil[*] — Sylvie Delaët[‡]

[†] Dept. of Computer Science, Kent State University, USA
[*] Université Pierre et Marie Curie-Paris 6, LIP6-CNRS, INRIA Grand Large, France

[‡] Université Paris Sud-XI, LRI-CNRS, France

N° 6529

May 2008

Thème NUM

Rapport de recherche

ISRN INRIA/RR--6529--FR+ENG

ISSN 0249-6399

Universe Detectors for Sybil Defense in Ad Hoc Wireless Networks

Adnan Vora[†] , Mikhail Nesterenko[†] , Sébastien Tixeuil[⋆] , Sylvie Delaët[‡]

[†] Dept. of Computer Science, Kent State University, USA
[⋆] Université Pierre et Marie Curie-Paris 6, LIP6-CNRS, INRIA Grand Large, France
[‡] Université Paris Sud-XI, LRI-CNRS, France

Thème NUM — Systèmes numériques
Projet Grand Large

Rapport de recherche n° 6529 — May 2008 — 28 pages

Abstract: The Sybil attack in unknown port networks such as wireless is not considered tractable. A wireless node is not capable of independently differentiating the universe of real nodes from the universe of arbitrary non-existent fictitious nodes created by the attacker. Similar to failure detectors, we propose to use *universe detectors* to help nodes determine which universe is real. In this paper, we (i) define several variants of the neighborhood discovery problem under Sybil attack (ii) propose a set of matching universe detectors (iii) demonstrate the necessity of additional topological constraints for the problems to be solvable: node density and communication range; (iv) present $\mathcal{SAND}$ — an algorithm that solves these problems with the help of appropriate universe detectors, this solution demonstrates that the proposed universe detectors are the weakest detectors possible for each problem.

Key-words: Sybil attack, wireles network, unverse detector

Unité de recherche INRIA Futurs
Parc Club Orsay Université, ZAC des Vignes,
4, rue Jacques Monod, 91893 ORSAY Cedex (France)
Téléphone : +33 1 72 92 59 00 — Télécopie : +33 1 60 19 66 08

Détecteurs d'Univers pour la Défense contre les Attaques Sybilles dans les Réseaux Ad Hoc Sans Fil

Résumé : Le problème de l'attaque Sybille dans les réseaux à ports inconus comme les réseaux sans fil n'est pas considéré comme soluble. Un nœud sans fil n'est pas capable de différencier par lui même un univers de nœuds réels d'un univers de nœuds fictifs créé par un attaquant.

De manière similaires aux détecteurs de défaillances, nous proposons d'utiliser des *détecteurs d'univers* pour aider les nœuds à déterminer quel univers est réel. Dans cet article, nous *(i)* définissons plusieurs variantes du problème de découverte de voisinage en présence d'attaques Sybilles; *(ii)* nous présentons un ensemble de détecteurs d'univers correspondants; *(iii)* nous prouvons la nécessité d'utiliser des contraintes topologiques supplémentaires pour que le problème devienne soluble: la densité des nœuds et la portée de communication; *(iv)* nous présentons $\mathcal{SAND}$, un algorithme distribué qui résoud les problèmes proposés à l'aide des détecteurs d'univers appropriés, et montrons que les détecteurs d'univers sont les plus faibles possibles pour chaque problème.

Mots-clés : Attaque Sybille, réseau sans fil, détecteur d'univers

1 Introduction

A Sybil attack, formulated by Douceur [9], is intriguing in its simplicity. However, such an attack can incur substantial damage to the computer system. In a Sybil attack, the adversary is able to compromise the system by creating an arbitrary number of identities that the system perceives as separate. If the attack is successful, the adversary may either overwhelm the system resources, thus channeling the attack into denial-of-service [25], or create more sophisticated problems, e.g. routing infrastructure breakdown [12].

Ad hoc wireless networks, such as a sensor networks, are a potential Sybil attack target. The ad hoc nature of such networks may result in scenarios where each node starts its operation without the knowledge of even its immediate neighborhood let alone the complete network topology. Yet, the broadcast nature of the wireless communication prevents each node from recognizing whether the messages that it receives are sent by the same or different senders. Thus, an attacker may be free to either create an arbitrary number of fictitious identities or impersonate already existing real nodes. The problem straddles the security and fault tolerance domains as the attacker may be either a malicious intruder or a node experiencing Byzantine fault. A fault is Buzantine [14] if the faulty node disregards the program code and behaves arbitrarily. For convenience, in this paper we assume that the attacker is a faulty node rather than intruder.

Problem motivation. A standard way of establishing trust between communicating parties is by employing cryptography. There is a number of publications addressing the Sybil attack in this manner [8, 16, 19, 23, 26, 27, 29]. For example, if each node has access to verified certificates and every sender digitally signs its messages, then the receiver can unambiguously determine the sender and discard superfluous identities created by the faulty node by checking the digital signature of the message against the certificates. However, there are several reasons for this approach to be inappropriate. A cryptography-based solution pre-supposes a key-based infrastructure which requires its maintenance and update and thus limits its applicability. Moreover, resource constrained devices, such as sensor nodes in sensor networks, may not be able to handle cryptographic operations altogether.

Another approach is intrusion detection based on *reputation* [1, 5, 11]. Due to the broadcast nature of wireless communication, the messages from each node are observed by its neighbors. A fault is detected if the node deviates from the protocol. It is unclear how reputation-based schemes would fare if the messages cannot be matched to the sender: the faulty node may impersonate other nodes

or create an arbitrary number of fictitious nodes and set up its own alternative reputation verification network.

However, there are two unique features of wireless communication that make defense against the Sybil attack possible. The wireless communication is broadcast. Thus, the message transmission of a faulty node is received by all nodes in its vicinity. In addition, the nodes can estimate the *received signal strength* (RSS) of the message and make judgments of the location of the sender on its basis. Note that the latter is not straightforward as the faulty node can change its *transmission signal strength* (TSS). In this paper we investigate the approaches to Sybil defense using this property of wireless communication.

Related literature. Newsome et al [18] as well as Shi and Perrig [22] survey various defenses against the Sybil attack. They stress the promise of the type of technique we consider. Demirbas and Song [7] consider using the RSS for Sybil defense.

A line of inquiry that is related to Sybil defense is secure location identification [2, 13, 15, 21, 24]. In this case, a set of trusted nodes attempt to verify the location of a possibly malicious or faulty node. However, the establishment of such trusted network is not addressed. Hence, this approach may not be useful for Sybil defense.

Delaët et al [6], and Hwang et al [10] consider the problem where the faulty node operates synchronously with the other nodes. Delaët et al [6] provided examples of positioning of faulty nodes and their strategies that lead to neighborhood discovery compromise. Note that the synchrony assumption places a bound on the number of distinct identities that the faulty node can assume before the correct nodes begin to counter its activities. Even though the faulty node may potentially create the infinite number of fictitious identities, the correct nodes have to deal with no more than several of them at a time. However, this approach simplifies the problem as it limits the power of the faulty node and the strength of the attack.

Nesterenko and Tixeuil [17] describe how, despite Byzantine faults, every node can determine the complete topology of the network despite once each node recognizes its immediate neighbors. Thus, to defend against the Sybil attack it is sufficient to locally solve Byzantine-robust neighborhood discovery.

Note that the problem is trivial when the ports are known. In this case, the receiver may not know the identity of the transmitter of the message but can match the same transmitter across messages. This prohibits the faulty node from creating more than a single fictitious identity or impersonating other real nodes and allows a simple solution.

Our approach and contribution. We consider the problem of neighbor identification in the presence of Byzantine nodes. The nodes are embedded in a geometric

plane and know their location. They do not have access to cryptographic operations. The nodes can exchange arbitrary messages, but the only information about the message that the receiver can reliably obtain is its RSS. We consider the asynchronous model of execution. That is, the execution speed of any pair of nodes in the network can differ arbitrarily. This enables the faulty node to create an arbitrary number of fictitious identities or impersonate the correct nodes in an arbitrary way. Moreover, in this model, the only unique identities that the nodes have are their coordinates. Hence, the objective of each node is to collect the coordinates of its neighbors. We focus on local solutions to the neighborhood discovery. That is, each node only processes messages from the correct neighbors within a certain fixed distance. We do not consider a denial-of-service attack or jamming attack [25], where the faulty nodes just overwhelm resources of the system by continuously transmitting arbitrary messages. We assume that the network has sufficient bandwidth for message exchanges and the nodes have sufficient memory and computing resources to process them. To the best of our knowledge, this is the most general model of Sybil defense considered to-date.

In Section 2 we provide details for our execution model and formally state several variants of the neighborhood discovery problem. Sections 3, 4, 5, and 6 outline the boundaries of the achievable. In Section 3, we formally prove that this problem is not solvable without outside help. Intuitively, the faulty node may create a *universe* of an arbitrary number of fictitious identities whose messages are internally consistent and the correct node has no way of differentiating those from the universe of correct nodes. In Section 4, we introduce *universe detectors* as a way to help nodes select the correct universe. The idea is patterned after failure detectors [4]. Just like failure detectors, universe detectors are not implementable in asynchronous systems. However, they provide a convenient abstraction that separates the concerns of algorithm design and implementation of the necessary synchrony and other details that enable the solution to Sybil defense. However, unlike failure detectors, universe detectors alone are insufficient to allow a solution to the neighborhood discovery problem. If the density of the network is too sparse, the faulty nodes may introduce a fictitious identity such that the detector is rendered unable to help the correct nodes. In Section 5, we prove the necessary condition for the location of the correct nodes to allow a solution to the neighborhood discovery problem. However, the faulty node may still be able to compromise the operation of correct nodes. For that, a faulty node may assume the identity of a correct node and discredit it by sending incorrect messages to other nodes. In Section 6 we prove another necessary condition for the minimum transmission range of correct nodes that eliminates this problem.

In Section 7 we present a Sybil-attack resilient neighborhood discovery algorithm $\mathcal{SAND}$ that uses the universe detectors to solve the neighborhood discovery problem provided that the necessary conditions are met. In their study of failure detectors Chandra et al [3] defined the weakest failure detector as the necessary detector to solve the problem that they are deployed to address. With the introduction of $\mathcal{SAND}$, we show that the employed detectors are the weakest detectors necessary to solve the neighborhood discovery problem. In Section 8, we conclude the paper by discussing the implementation details of the algorithm and the attendant universe detectors.

2 Computation Model Description, Assumptions, Notation and Definitions

A computer network consists of nodes embedded in a geometric plane. Each node is aware of its own coordinates. A (*node*) *layout* is a particular set of nodes and their locations on the plane. Unless explicitly restricted, we assume that the node layout can be arbitrary. Any specific point on the plane can be occupied by at most one node. Thus, the node's coordinates on the plane uniquely identify it. The nodes have no other identifiers. For ease of exposition, we use identifiers at the end of the alphabet such as u or v to refer to the particular locations or non-faulty nodes occupying them. We use f and k respectively to refer to a faulty node and a location where the faulty node may pretend to be located. The distance between u and v is $|uv|$. The *neighborhood set* or just *neighborhood* of a node u is a set of nodes whose distance to u is less than a certain fixed distance d_n.

Program model. We assume the asynchronous model of algorithm execution. That is, the difference between the execution speed of any pair of nodes can be arbitrarily large. Note that this asynchrony assumption allows any node, including a faulty one, to send an arbitrary number of messages before other nodes are able to respond. The nodes run a distributed algorithm. The algorithm consists of variables and actions. A (*global*) *state* of the algorithm is an assignment of values to all its variables. An action is *enabled* in a state if it can be executed at this state. A *computation* is a maximal fair sequence of algorithm states starting from a certain prescribed initial state s_0 such that for each state s_i, the next state s_{i+1} is obtained by atomically executing an action that is enabled in s_i. Maximality of a computation means that the computation is either infinite or terminates where none of the actions are enabled. In other words, a computation cannot be a proper prefix of another computation. Fairness means that if an action is enabled in all but finitely many

states of an infinite computation then this action is executed infinitely often. That is, we assume *weak fairness* of action execution. During a single computation, node layout is fixed.

Nodes can be either correct or faulty (Byzantine). A faulty node does not have to follow the steps of the algorithm and can behave arbitrarily throughout the computation.

Node communication. Nodes communicate by broadcasting messages. As the distance to the sender increases, the signal fades. We assume the free space model [20] of signal propagation. The antennas are omnidirectional. The received signal strength (RSS) changes as follows:

$$R = cT/r^2 \tag{1}$$

where R is the RSS, c is a constant, T is the transmitted (or sent) signal strength (TSS), and r is the distance from the sender to the receiver. We assume that r cannot be arbitrarily small. Thus, R is always finite. There is a minimum signal strength R_{min} at which the message can still be received. There is no message loss. That is, if a message is sent with TSS — T', then every node within distance $r' = \sqrt{cT'/R_{min}}$ of the sender receives the message. We do not consider interference, hidden-terminal effect or other causes of message loss. We assume that every correct node always broadcasts with a certain fixed strength T_r. A *range* r_t is defined as $\sqrt{cT_r/R_{min}}$. The relation between range r_t and neighborhood distance d_n is, in general, arbitrary. A faulty node may select arbitrary TSS. If a node receives a message (i.e. if the RSS is greater than R_{min}), then the node can accurately measure the RSS.

To simplify the exposition we assume that the nodes transmit three types of messages: (i) u transmits *announce*, this message has only the information about u and carries u's coordinates; the purpose of an announcement is for u to advertise its presence to its neighbors; (ii) u transmits *confirm* of another node v's transmission; (iii) u transmits *conflict* with another node v's transmission if its observations do not match the location or the contents of v's message. The original message is attached in *confirm* and *conflict*. Every message contains the coordinates of the sender.

Fictitious nodes and conflicts. Since the only way to unambiguously differentiate the nodes is by their location, the objective of every node is to determine the coordinates of its neighbors. Faulty nodes may try to disrupt this process by making the correct node assume that it has a non-existent neighbor. Such a non-existent neighbor is *fictitious*. A node that indeed exists in the layout is *real*. Note that a real node can still be either correct or faulty. Faulty nodes may try to tune their TSS and otherwise transmit messages such that it appears to the correct nodes that

the message comes from a fictitious node. Moreover, the faulty nodes may try to make their transmissions appear to have come from correct nodes.

As a node receives messages, due to the actions of a faulty node, the collected information may be contradictory. A *conflict* consists of a message of any type purportedly coming from node k, yet the received signal strength at node u does not match $|uk|$ provided that the signal were broadcast from k with the TSS of T_r. A conflict is *explicit* if u receives this conflicting message. Note that the RSS may be so low that u is unable to receive the message altogether, even though the RSS at u should be greater than R_{min} in case the message indeed come from k and be broadcast at T_r. In this case the conflict is *implicit*. To discover the implicit conflict u has to consult other nodes that received the conflicting message. If u detects a conflict it sends a conflict message.

A *universe* is a subset of neighbors that do not conflict. That is, a universe at node u contains nodes v and w whose announcements u received such that u did not receive a conflict from v about w or from w about v. Note that due to conflicts the information collected by a single node may result in several different universes. A universe is *real* if all nodes in it are real. A universe is *complete* for a node u if it contains all of u's correct neighbors. Note that even though a faulty node is real, it can evade being added to universes by not sending any messages. Hence, a complete universe is not required to contain all the real nodes, just correct ones.

Program locality. To preserve the locality of a solution to the neighborhood discovery problem, we introduce the following requirement. Each node ignores information from the nodes outside the range r_t and about the nodes outside the neighborhood distance d_n.

Problem statement. We define several variants of the problem. The *strong neighborhood discovery problem* $\mathcal{SNDP}$ requires each correct node u to output its neighborhood set according to the following properties:

safety — if the neighborhood set of u is output, the set contains only all correct nodes and no fictitious nodes of u's neighborhood;

liveness — every computation has a suffix in whose every state u outputs a neighborhood set that contains all correct neighbors of u. In other words, u eventually outputs its complete neighborhood set.

This problem definition may be too strict. Some correct nodes may be slow in announcing their presence. However, the safety property of $\mathcal{SNDP}$ requires each node to wait for its slow neighbors before outputting the neighborhood set. Hence,

we define the *weak neighborhood discovery problem* $\mathcal{WNDP}$. This problem relaxes the safety property to allow the output neighborhood set to contain a subset of correct neighbors of u. Note that the presence of the fictitious nodes in the output is still prohibited. Also note that the liveness property requires that the neighborhood set of u in $\mathcal{WNDP}$ eventually contains all correct neighbors. Further relaxation of the safety property yields the *eventual neighborhood discovery problem* $\diamond\mathcal{NDP}$. It requires that the safety of $\mathcal{SNDP}$ be satisfied only in the suffix of a computation. That is $\diamond\mathcal{NDP}$ allows the correct nodes to output incorrect information arbitrarily long before providing correct output. Observe that any solution to $\mathcal{SNDP}$ is also a solution to $\mathcal{WNDP}$, and any solution to $\mathcal{WNDP}$ is also a solution to $\diamond\mathcal{NDP}$.

3 Impossibility of Standalone Solution to Neighborhood Discovery

In this section we demonstrate that in the asynchronous system any correct node is incapable of discovering its neighborhood if a faulty node is present. The intuition for this result is as follows. Since a faulty node is not restricted in the number of messages that it generates, it can send an arbitrary number of announcements introducing fictitious nodes. The faulty node can then imitate arbitrary message traffic between these non-existent nodes. On its own, a correct node is not able to differentiate these fictitious nodes from the real ones.

Theorem 1 *In an asynchronous system, none of the three variants of the neighborhood discovery problem are deterministically solvable in the presence of a single Byzantine fault.*

Proof: We provide the proof for the eventual neighborhood discovery problem. Since this problem is the weakest of the three that we defined, the impossibility of its solution implies similar impossibility for the other two.

Assume the opposite. Let $\mathcal{A}$ be a deterministic algorithm that solves $\diamond\mathcal{NDP}$ in the presence of a faulty node. Let us consider an arbitrary layout L_1 that contains a faulty node f. Let us consider another layout L_2 containing f such that the neighborhood U_1 in layout L_1 of at least one correct node u differs from its neighborhood U_2 in L_2 and this difference includes at least one correct node. Without loss of generality we can assume that there exists a correct node v such that $v \in U_1$ and $v \notin U_2$.

We construct two computations of $\mathcal{A}$: σ_1 on layout L_1 and σ_2 on layout L_2. The construction proceeds by iteratively enlarging the prefixes of the two computations.

In each iteration, we consider the last state of the prefix of σ_1 constructed so far and find the action that was enabled for the longest number of consequent steps. If there are several such actions, we choose one arbitrarily. We attach the execution of this action to the prefix of σ_1. If this action is a message transmission of a node w such that $w \in U_1$, we also attach the following action execution to the prefix of σ_2: node f sends exactly the same message as w in σ_1 with the TSS selected as $T = T_r|uf|^2/|uw|^2$. Observe that u receives the same message and with the same RSS in this step of σ_2 as in the step added to σ_1. If the new action attached to σ_1 prefix is not by a node in U_1, or it is not a message transmission, no action is attached to the prefix of σ_1. We perform similar operations to the prefix of σ_2.

We continue this iterative process until maximal computations σ_1 and σ_2 are obtained. Observe that by construction, both computations are weakly fair computations of $\mathcal{A}$. Moreover, in both cases u receives exactly the same messages with exactly the same RSS.

By assumption, $\mathcal{A}$ is a solution to $\diamond\mathcal{NDP}$. According to the liveness property of the problem, σ_1 has a suffix where u outputs its neighborhood in every state and, due to the liveness property, σ_1 contains a suffix where u's neighborhood set contains all correct nodes. In layout L_1 of σ_1, v is u's correct neighbor. Hence, v has to be included in this set. That is, there is a suffix of σ_1 where u outputs a neighborhood set that contains v. However, u receives the same messages in σ_2. Since $\mathcal{A}$ is deterministic, u has to output exactly the same set in σ_2 as well. That is, σ_2 contains a suffix where the neighborhood set also contains v. However, v is fictitious in layout L_2 of σ_2. According to the safety property of $\diamond\mathcal{NDP}$, every computation should contain a suffix where the neighborhood set of u excludes fictitious nodes. That is, σ_2 of $\mathcal{A}$ violates the safety of $\diamond\mathcal{NDP}$. Hence, our assumption that $\mathcal{A}$ is a solution to the weak neighborhood discovery problem is incorrect. The theorem follows. □

4 Abstract Universe Detectors

Definitions. The negative result of Theorem 1 hinges on the ability of a faulty node to introduce an arbitrary number of fictitious nodes. A correct node cannot distinguish them from its real neighbors. Still, a correct node may be able to detect conflicts between nodes and separate them into universes. However, it needs help deciding which universe is real. This leads us to introduce the concept of a universe detector that enables the solution to the neighborhood discovery problem in the asynchronous computation model. A *universe detector* indicates to each correct

node which universe is real. It takes the universes collected by the node as input and outputs which universe contains only real nodes. That is, a universe detector *points* to the real universe. Note that the algorithm still has to collect the neighborhood information and separate them into universes such that at least one of them is real. If the algorithm does not provide a real universe, the detector does not help.

Depending on the quality of the output, we define the following detector classes. For each node u, a *strongly perfect universe detector* $\mathcal{SPU}$ has the following properties:

completeness — if a computation contains a suffix where in every state, u outputs a real and complete universe, then this computation also contains a suffix where $\mathcal{SPU}$ at u points to it;

accuracy — if $\mathcal{SPU}$ points to a universe, this universe is real and complete.

The strongly perfect universe detector may be too restrictive or too difficult to implement. Unlike $\mathcal{SPU}$, a *weakly perfect universe detector* $\mathcal{WPU}$ may point to a real universe even if it is not complete. That is, the definition of accuracy is relaxed to allow the detector to point to a real universe that is not complete. Note that $\mathcal{WPU}$ still satisfies the completeness property and has to eventually point to the real universe if it is available. A further relaxation of completeness and accuracy yields an *eventually perfect universe detector* $\diamond\mathcal{PU}$ which satisfies both properties in a suffix of every computation. Observe that the relationship between these detector classes is as follows: $\mathcal{SPU} \subset \mathcal{WPU} \subset \diamond\mathcal{PU}$

Observe that these universe detectors enable a trivial solution to the neighborhood discovery problems: each node composes a universe for every possible combination of the nodes that claim to be in its neighborhood. Naturally, as the node receives announcements from all its correct neighbors, one of these universes is bound to be real and complete. Hence, the detector can point to it. However, such an approach essentially shifts the burden of separating fictitious and real nodes to the detector while we are interested in minimizing the detector's involvement. This leads us to introduce an additional property of the algorithms that we consider. An algorithm that solves the neighborhood discovery problem that uses detectors is *conflict-aware* if for each universe U of node u, if nodes v and w do not have a conflict and v belongs to U then w also belongs to U. That is, the algorithm does not gratuitously separate non-conflicting neighbors into different universes. In what follows we focus on conflict-aware solutions.

5 Necessary Node Density

Theorem 1 demonstrates that to solve the neighborhood discovery problem, any algorithm requires outside help from a construct like a universe detector. However, the availability of a universe detector may not be sufficient. Faulty nodes may take advantage of a layout to announce a fictitious node without generating conflicts. Then, a correct node running a conflict aware algorithm never removes this fictitious node from the real universe. A universe detector then cannot point to such a universe.

To illustrate the idea we start with a sequence of fictitious node placement examples.

5.1 Fictitious Nodes Placement Examples

For this discussion we consider the neighborhood of a certain correct node u and a faulty node f that tries to compromise u's neighborhood discovery. We denote x, y, z — the correct nodes in the neighborhood of u that that are respectively first, second and third nearest to f. Note that to affect u, the faulty node f does not itself have to be the neighbor of u. Our analysis proceeds according to the number of correct receivers of messages sent by f.

Single correct receiver. Refer to Figure 1 for illustration. Note that due to the broadcast nature of radio signal propagation, if any correct node receives a message sent by f, x also receives this message because it is closest to f. Therefore, the single correct receiver may only be x. Note, that for y to not receive the signal from f, the transmission signal strength should be sufficiently low. Recall that a correct node always broadcasts with pre-defined signal strength T_r. Thus, to deceive x, f has to select the location of k and the TSS such that: (i) the RSS at x is the same as if k transmitted with T_r and (ii) the RSS at y is below R_{min}. For the received signal strength at y to be less than R_{min}, k cannot be closer to x than $|fy|$. On the other hand, the location of k cannot be outside the range r_t (or else x generates conflict) or outside the neighborhood distance d_n (or else x ignores it). Thus, the possible location of k is a ring around x with the inner radius $|fy|$ and the outer radius — $min(r_t, d_n)$.

Two correct receivers. If there are exactly two correct receivers, they are the two nodes x and y nearest to f. Assume that f makes two transmissions at signal strengths T_1 and T_2. For these transmissions, the RSS at x and y are R_{x1}, R_{y1} and

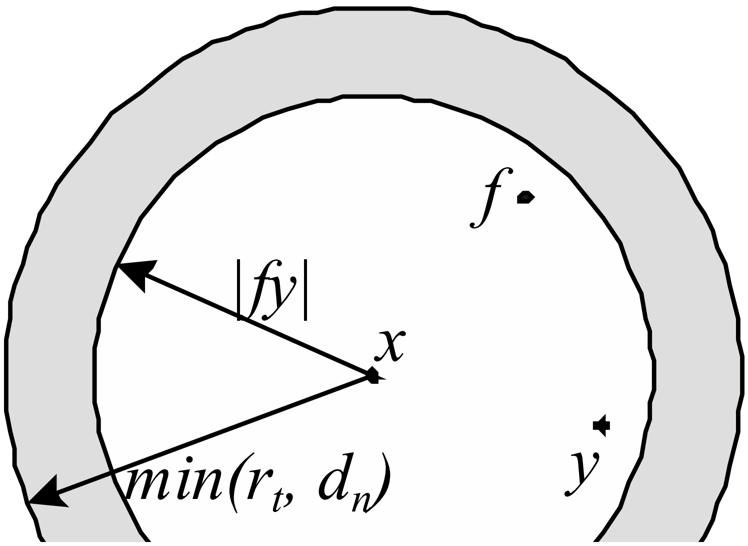

Figure 1: Deception field with a single correct node x.

R_{x2}, R_{y2} respectively. From the signal attenuation in Formula 1 we obtain:

$$\frac{|fy|}{|fx|} = \sqrt{\frac{R_{x1}}{R_{y1}}} = \sqrt{\frac{R_{x2}}{R_{y2}}}$$

That is, regardless of transmission power, the ratio of received signal strengths at x and y does not change. Hence, f may select the location of the fictitious node k such that it preserves this ratio. Such locations form an arc of a circle. Refer to Figure 2. The center of the circle lies on the line whose segment is (xy). The radius of the circle is $ba/(b-a)$ where b and a are the portions of (xy) such that $b/a = |fy|/|fx|$. This circle is the *deception circle*. Note that f may not be able to use all of the deception circle for fictitious node placement: to get both x and y to receive the signal without generating conflicts the points on the arc have to lie within $min(r_t, d_n)$ of both x and y. Moreover, similar to the single-receiver case, the portion of the arc that is closer to y than $|fz|$ cannot be used without z also receiving the message.

More than two correct receivers. Note that if there are more than two correct receivers, they can be considered pairwise. Each pair of correct receivers forms its own deception circle. Note that k can only be placed at the intersection of all

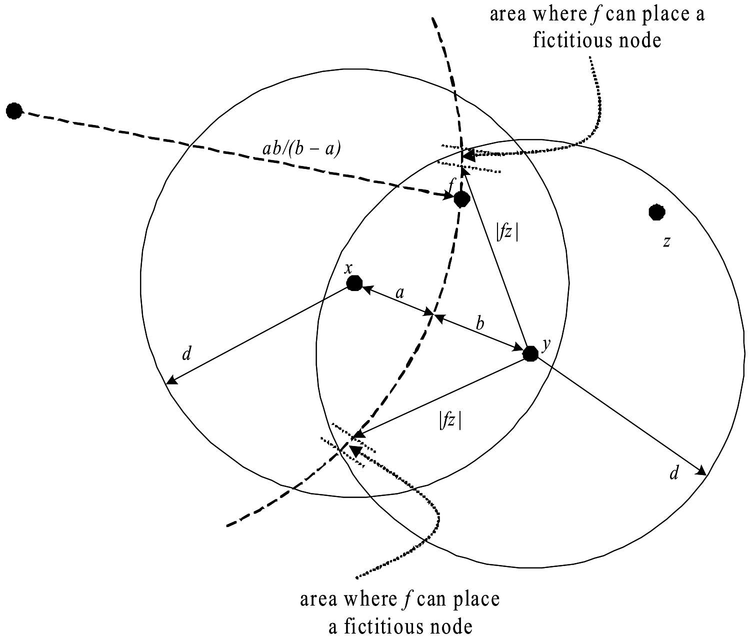

Figure 2: Deception field with a two-node retinue. Correct nodes x and y receive transmissions of faulty node f, while z does not.

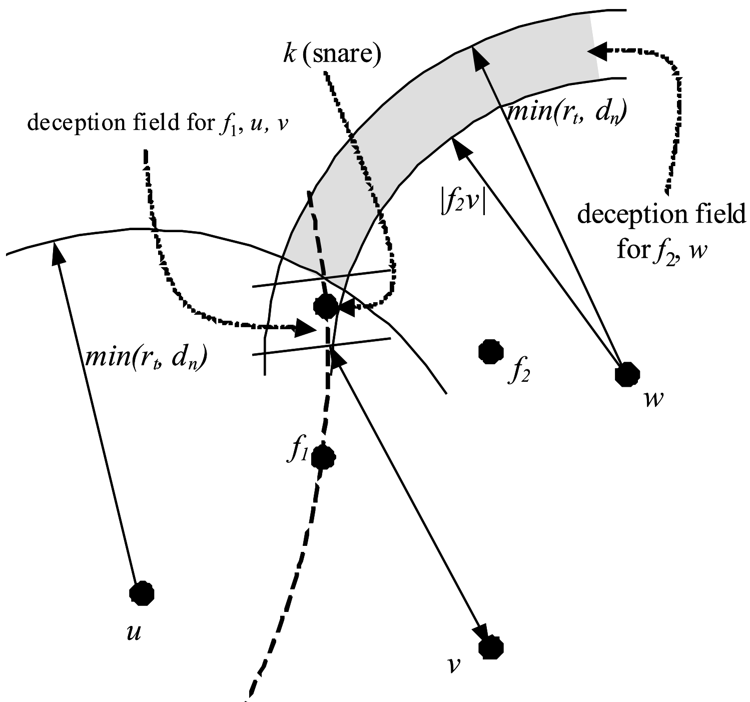

Figure 3: The location of a snare in case of multiple faulty nodes. The retinue of f_1 is x and y. The retinue of f_2 is z. The intersection of deception fields produces area where a snare can be placed.

these circles. Note, however, that the circles intersect in the same place only if the recipients are co-linear.

Snare. A faulty node may affect the correct nodes around it. A set E_f of correct nodes is the *retinue* of a faulty node f if the following holds: if a correct node u belongs to E_f, then every correct node v such that $|vf| \leq |uf|$, also belongs to E_f. The faulty node is the *leader* of the retinue. For example, assume there are two faulty nodes f_1 and f_2 and three correct nodes u, v and w such that $|f_1u| < |f_1v| < |f_1w|$ and $|f_2w| < |f_2v| < |f_2u|$. Refer to Figure 3 for illustration. All three correct nodes can be either in the retinue E_{f1} of f_1 or E_{f2} of f_2. However, if v belongs of E_{f1}, so does u, and if u belongs to E_{f2}, so do v and w.

A *deception field* for a retinue of a faulty node f is the area such that for each point k of the field there exists a TSS that the leader of the retinue can use to transmit a message. The message so transmitted generates the RSS at each member of the retinue as if the message was sent from k with transmission strength T_r. Intuitively, a deception field is the area where f can place fictitious nodes without generating conflicts at its retinue members.

A point k in a neighborhood of a correct node u is a (*simple*) *snare* for u if there exists a set of faulty nodes and a retinue assignment for them such that: u is in one of the retinues and the intersection of the deception fields of the retinues includes k. Note that the nodes in range of k are either in the retinues or not. Intuitively, a snare is a point where faulty nodes can jointly place a fictitious node without generating explicit conflicts at any of the correct neighbors of u. Refer to Figure 3 for illustration. Note that some of the nodes may have implicit conflicts with k. That is, they are within range r_t of k and u but not in one of the retinues. That is, they should receive a message from a node at k but they do not. Note that a snare transmission from faulty nodes may still generate conflicts outside the range of u. However, due to the locality assumption, u ignores this conflict.

A point k is a *perfect snare* for u if it is a snare and all nodes within the transmission range of u and k are in the retinues of the faulty nodes participating in the snare. That is, if faulty nodes broadcast in a perfect snare, neither explicit nor implicit conflicts are generated at the neighbors of u.

Evaluating fault tolerance of a layout. To illustrate the concept of a snare, let us discuss a square grid layout (refer to Figure 4). Let s be the distance between the nodes in the grid. Note that for a node to have any neighbors, the neighborhood distance d_n has to be no less than s. Let $s \leq d_n < s\sqrt{2}$. For simplicity, let $r_t = d_n$. Then each node has exactly four neighbors. Note that the failure of a single node

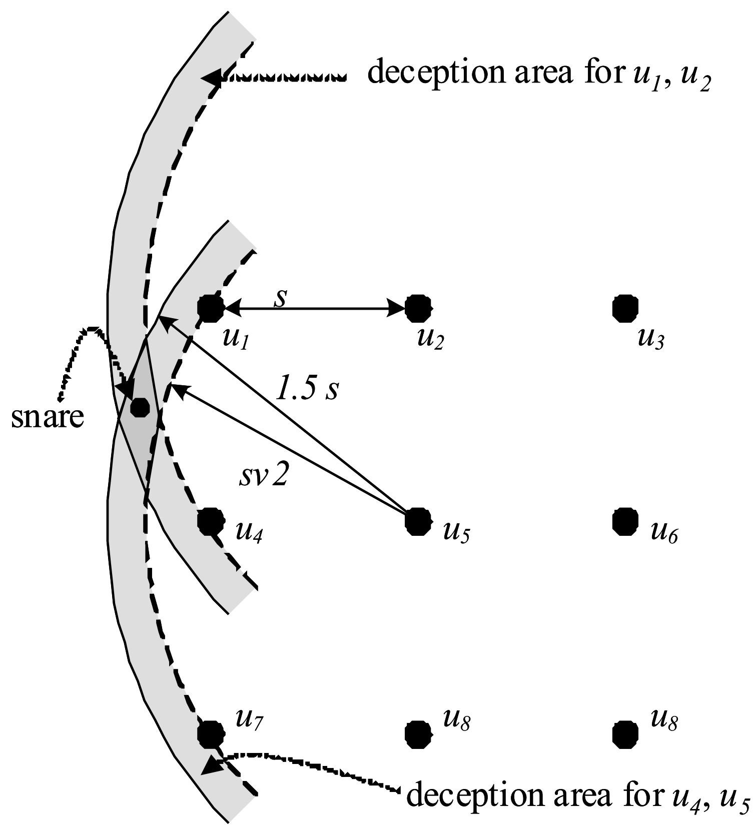

Figure 4: The possibility of a snare in a grid layout with $d_n = 1.5s$.

creates a wedge-shaped deception field around the faulty node. Thus, with this distance the layout is not fault tolerant.

Let us consider the case where $s\sqrt{2} \leq d_n < 2s$ and again $r_t = d_n$. In this case, the neighborhood can withstand a failure of exactly one node. Indeed, assume that a single node failed. Note that we have to consider the collusion of this faulty node with arbitrary faulty nodes outside the neighborhood. Let us focus on the neighborhood of node u_5. For u_5 to consider a fictitious node, the transmission of at least one faulty node has to reach u_5. However, if a signal from a faulty node, either inside or outside the neighborhood of u_5, reaches u_5, then this signal is received by at least two more correct neighbors of u_5. Moreover, the three correct nodes that receive this signal are non-collinear. This means that their pairwise deception circles intersect only in the sender itself. Thus, the neighborhood of u_5 does not contain a snare.

Let us determine if this grid layout can withstand simultaneous failure of two nodes in the same neighborhood. Let $d_n = r_t$ be $1.5s$. Suppose nodes u_1 and u_4 fail. The deception field of u_1 with u_2 in its retinue is a disk with outer circle radius $1.5s$ and inner — $s\sqrt{2}$. That is, the outer disk is the range for correct nodes $r_t = d_n$ and the inner is the distance to the next nearest node — u_5. A similar disk is a deception field of u_4 for u_5. The intersection of the two disks forms the area where a perfect snare for u_5 may be located. To use the snare, u_1 sends messages to u_2, and u_4 to u_5 with the appropriate TSS pretending that the messages come from a fictitious node located in the snare. Thus, the grid layout with such d_n cannot withstand a two-node failure.

5.2 Necessary Node Density Condition

Having described the required instruments, we now demonstrate that the availability of the universe detectors alone is not sufficient to enable a solution to any of the neighborhood discovery problems if the node layout is too sparse. That is, if the nodes are not properly positioned on the plane.

To simplify the proof we consider solutions that are *well-formed*. An algorithm is well-formed if (i) the action that transmits *announcement* is always enabled until executed; (ii) the receipt of a message may enable either *confirm* or *conflict*, this action stays enabled until executed.

Theorem 2 *There is no conflict-aware well-formed deterministic solution to any of the neighborhood discovery problems despite the availability of the universe detectors if one of the considered layouts contains a perfect snare.*

Proof: In the proof, we focus again on the weakest of the problems: the eventual neighborhood discovery. Assume the opposite: there is a conflict-aware well-formed algorithm $\mathcal{A}$ that uses a detector and solves the problem even though in one of the layouts L_1, the neighborhood of a correct node u contains a perfect snare k.

Consider a layout L_2 that is identical to L_1 except that there is a correct node at location k in L_2. We construct a computation σ_2 of $\mathcal{A}$ on L_2 as follows. Faulty nodes do not send any messages in σ_2. We arrange the neighbors of u, including u itself, into an arbitrary sequence Q. We then build the prefix of σ_2 by iterating over this sequence. Since $\mathcal{A}$ is well-formed, each node in the sequence has *announcement* enabled. We add the action execution that transmits *announcement* to σ_2 in the order of nodes in Q. Since $\mathcal{A}$ is well-formed, these transmissions may enable *confirm* actions at the neighbors of u. Note that since v is correct, *conflict* actions are not enabled by these transmissions. We now iterate over the nodes in Q. For each node v we add the execution of these *confirm* actions at v to σ_2 in arbitrary fixed order, for example in the order that the original senders the appear in Q. We proceed in this manner until the sequence Q is exhausted. Note that these transmissions may potentially generate another round of *confirm* messages at the nodes in Q. We continue iterating over Q until no more messages are generated. We then complete σ_2 by executing the actions of nodes in an arbitrary fair manner. Note that the remaining messages deal with the nodes outside u's neighborhood. Therefore, u ignores them.

Now, the liveness property of all the detectors states that a detector points to a universe if it is output for a suffix of the computation. Since $\mathcal{A}$ is a solution of $\diamond\mathcal{NDP}$ and σ_2 is a computation of $\mathcal{A}$, σ_2 has to contain a suffix where u outputs a real universe in every state. Since k is a correct neighbor of u, k is included in the real universe.

Recall that in layout L_1, point k is a perfect snare. This means that there is an arrangement of retinues and the TSS for the faulty nodes, such that when the faulty nodes transmit, each node in the neighborhood of u in the distance d from k receives a message with the same RSS as if a node at k broadcast with T_d. Moreover, none of the nodes in the neighborhood of u detect conflicts.

We construct a computation σ_1 of $\mathcal{A}$ on layout L_1 as follows. We iterate over the same sequence Q as in σ_2. Note that k is also present in the sequence even though it is fictitious in σ_1. To build the prefix of σ_1 we execute similar actions as for σ_2. The only difference is that when node k broadcasts in σ_2, in σ_1 we have the faulty nodes that constitute the snare broadcast at the appropriate TSS. Note that in the computation thus formed, the correct neighbors of u receive messages at the

same RSS and with the same content from the faulty nodes as in σ_2 from k. Thus, these transmissions do not generate conflicts. Observe that this means that node u receives the same messages with the same RSS, and in the same sequence in σ_1 and σ_2. Since $\mathcal{A}$ is deterministic, u has to output the same universes in σ_1 and σ_2. Note also, that this means that u does not record conflicts. Since $\mathcal{A}$ is conflict aware, all u's universes of $\mathcal{A}$ include k together with the correct neighbors.

However, k is a fictitious node in L_1. This means that σ_1 contains a suffix where u does not output a real universe. According to the safety property of the detectors, none of them provides output in a suffix of σ_1. Which means that $\mathcal{A}$ does not output a neighborhood set in a suffix of σ_1. This violates the liveness property of a solution to $\diamond\mathcal{NDP}$. Therefore, our assumption that $\mathcal{A}$ is a solution to $\diamond\mathcal{NDP}$ is incorrect. The theorem follows. □

6 Necessary Transmission Range

In this section we provide another required condition for the existence of a solution to the neighborhood discovery problem. Essentially, if the nodes in the same neighborhood are out of range, the faulty node may introduce a conflict between them. This forces the algorithm to mistakenly split the correct nodes into separate universes and renders the failure detector powerless.

Theorem 3 *There is no conflict-aware deterministic solution for any of the neighborhood discovery problems despite the availability of universe detectors and lack of snares if the node transmission range r_t is less than double the neighborhood distance d_n.*

Proof: Consider the eventual neighborhood discovery and assume that there is an algorithm $\mathcal{A}$ that solves the problem in the presence of detectors on any layout without snares yet the transmission range of the correct nodes r_t is less than $2d_n$. Consider the layout L_1 where the neighborhood of a correct node u contains two nodes v and f_1 as well as a point k with the following properties. Refer to Figure 5 for illustration. As usual, v is correct, f_1 is faulty and there is no node at point k. Even though point k is in the neighborhood of u, it is out of range of v. That is, $r_t < |vk|$. Recall that this is possible since, by assumption, $r_t < 2d_n$. Node f_1 is such that $|uf_1| = |uk|$ and $r_t > |vf_1|$. The rest of the correct nodes in range of u are located such that, with the exception of v, k forms a perfect snare for u. That is, if f_1 sends a message from a fictitious node k, the only node that generates conflict

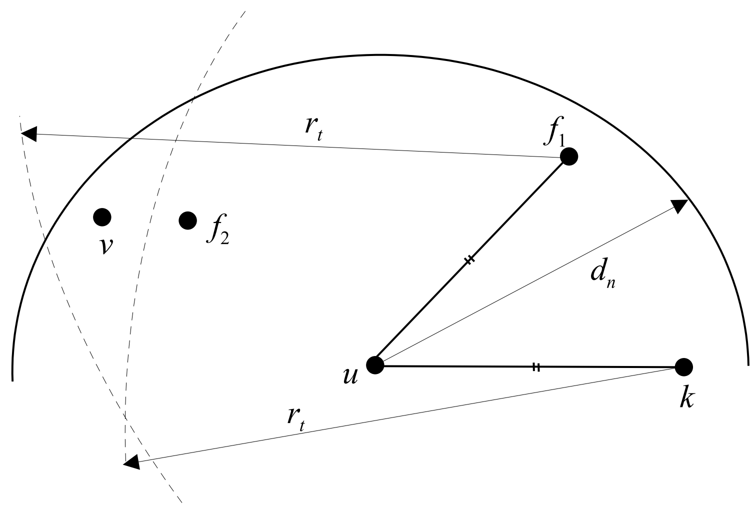

Figure 5: Insufficient range for recognition of faulty node. Illustration to the proof of Theorem 3.

is v. Certainly, with the presence of v, k is not a snare so the assumptions of the theorem apply.

Consider that f_1 indeed sends *announcement* pretending to be a fictitious node at k. Nodes f_1 and k are equidistant from u. Thus, if f_1 does not want u to detect a conflict, f_1 has to send the signal with the TSS of T_r. However, with such TSS, v is in range of f_1 but out of range of k. This means that v receives the announcement ostensibly coming from k and detects a conflict. The RSS at v is $cT_r/|vf_1|^2$. Since $\mathcal{A}$ is a solution to the neighborhood discovery problem and v is the only node that is aware of the conflict, v has to send *conflict* to u which removes the fictitious node k from the real universe of u.

Consider a different layout L_2 (refer to Figure 5) which is similar to L_1, only point k is occupied by a correct node and there is a faulty node f_2 near v. Specifically, the distance $|vf_2|$ is such that there are no correct nodes within the following range of f_2:

$$\frac{|vf_2|}{|vf_1|}\sqrt{\frac{c}{R_{min}}}$$

This ensures that when f_2 is going to imitate node k, none of the nodes besides v receive the messages from f_2. Note that f_2 and k still do not form a snare because

v is aware of the conflict. Note also, that such location of f_2 can always be found if the faulty node can be placed arbitrarily close to v.

Assume that if the node k in L_2 sends a message, f_2 replicates this message with TSS

$$\frac{T_r|vf_2|^2}{|vf_1|^2}$$

Observe that in this case all nodes, including v and u, receive exactly the same messages as in layout L_1. Since $\mathcal{A}$ is deterministic, the nodes have to act exactly as in the previous case. That is, v has to issue a conflict with the message of node k. However, after receiving this conflict, k is separated from u's real universe. Recall that k is correct in layout L_2. Note that in this case k is never going to be added to the output of $\mathcal{A}$ at u. However, this violates the liveness property of the neighborhood discovery problem since k is a correct neighbor of u. Thus, $\mathcal{A}$ is not a solution to this problem as we initially assumed. □

7 The Sybil Attack Resilient Neighborhood Discovery Algorithm $\mathcal{SAND}$

Our description of the algorithm proceeds as follows. We first motivate the need to frugally encode the universes to be passed to the universes detectors. We then describe the operation of the neighborhood detection algorithm itself. Then, we define the concrete implementations of the abstract detectors specified in Section 4. These concrete detectors should operate with our algorithm. On the basis of the algorithm and detector description we state the theorem of algorithm correctness and detector optimality.

Encoding universes. Observe that a naïve solution for representing universes by the algorithm results in an exponential number of universes. Indeed, assume that node u compiled a set of nodes U that do not conflict with two nodes v and w. Suppose now that u records a conflict between the two nodes. They thus have to be placed in separate universes: $U \cup \{v\}$ and $U \cup \{w\}$. Let us consider another pair of conflicting nodes x and y that are different from v and w. Then, there are four possible universes: $U \cup \{vx\}$, $U \cup \{vy\}$, $U \cup \{wx\}$, and $U \cup \{wy\}$. Hence, if there are N nodes in the neighborhood of u, the potential number of conflicting pairs is $\lfloor N/2 \rfloor$ and the number of universes is $2^{\lfloor N/2 \rfloor}$.

Therefore, our algorithm encodes the universes in the conflicts that are passed to the detector. That is, the algorithm passes a set of conflicts for the detector to generate the appropriate universe on its own.

Recall also that in an asynchronous radio network the receiving node can not distinguish one sender from another or decide if the two messages were sent by the same node. This task has to be handled by the detector.

Algorithm description. We assume that the necessary conditions for the existence of a solution to the neighborhood discovery problem are satisfied: the layout does not contain a (simple) snare and transmission range is at least twice as large as the neighborhood distance d_n.

The $\mathcal{SAND}$ algorithm operates as follows. Every message transmitted by the node contains its coordinates. Each node sends *announce*. After receiving an *announce*, a node replies with a *confirm* message. Each *confirm* contains the information of the announcement. If a node receives a message whose coordinates do not match the received signal strength, the node replies with a *conflict* message. The *conflict* also contains the information of the message that generated the conflict. Observe that *confirm* can only be generated by *announce* while *conflict* can be generated by an arbitrary message. Note that according to the locality assumption every node ignores messages from the nodes outside of its neighborhood distance d_n.

Each node u builds a message dependency directed graph *DEP*. For each *confirm*, u finds a matching *announce*; for each *conflict* — a matching message that caused the conflict. Note that this message dependence may not be unique. For example a faulty node may send a message identical to a message sent by a correct node. Since a node cannot differentiate senders in asynchronous radio networks, identical messages are merged in *DEP*. Note also, that a match may not be found because the faulty node may send a spurious conflict message or the conflict message is in reply to the faulty node message that u does not receive. Node u removes the unmatched message. Also, u removes the cycles and sinks of *DEP* that are not *announce*. Observe that *DEP* may grow indefinitely as faulty nodes can continue to send arbitrary messages.

Due to no-snare and transmission range assumptions, for every correct process u the following is guaranteed about *DEP*:

- Eventually, u receives an announcement from every correct node in its neighborhood. An announcement from each correct node will be confirmed by every correct node. There will be no messages from the correct nodes that conflict with any other messages from the correct nodes.

- Eventually, every message from a fictitious node will be followed up by at least one *conflict* message sent by one of the correct nodes from the neighborhood of u.

Concrete universe detectors. We define the *concrete* detectors $c\mathcal{SPU}$, $c\mathcal{WPU}$ and $\diamond c\mathcal{PU}$ as the detectors that accept the *DEP* provided by $\mathcal{SAND}$ as input and whose output satisfies the specification of the corresponding abstract detectors described in Section 4. That is, for each correct node u, $c\mathcal{SPU}$ only outputs complete and real universe, $c\mathcal{WPU}$ may output a real universe that is not complete, while $\diamond c\mathcal{PU}$ may provide arbitrary output for a fixed number of computation states. However, all three detectors eventually output the complete and real universe for u. Observe that the detectors have to comply with the specification even though *DEP* may grow infinitely large.

In $\mathcal{SAND}$, each process u observes the output of the detector and immediately outputs the universe presented by the detector without further modification. By the construction of $\mathcal{SAND}$ proves the following theorem.

Theorem 4 *Considering layouts without simple snares and assuming that the transmission range is at least twice as large as the neighborhood distance, the Sybil Attack Neighborhood Detection Algorithm $\mathcal{SAND}$ provides a conflict-aware deterministic solution to the Neighborhood Discovery Problem as follows: $\mathcal{SNDP}$ if $c\mathcal{SPU}$ detector is used; $\mathcal{WNDP}$ if $c\mathcal{WPU}$ is used; and $\diamond\mathcal{NDP}$ if $\diamond c\mathcal{PU}$ is used.*

Similar to Chandra et al [3] we can introduce the concept of a weakest universe detector needed to solve a certain problem. A universe detector $\mathcal{U}$ is the *weakest* detector required to solve a problem $\mathcal{P}$ if the following two properties hold:

- there is an algorithm $\mathcal{A}$ that uses $\mathcal{U}$ to solve $\mathcal{P}$;
- there is another algorithm $\mathcal{B}$ that uses the input of an arbitrary solution $\mathcal{S}$ of $\mathcal{P}$ to implement $\mathcal{U}$.

That is, $\mathcal{B}$ uses the output of $\mathcal{S}$ and provides the computations expected of $\mathcal{U}$. The intuition is that if any solution can be used to implement $\mathcal{U}$, then every solution needs the strength of at least $\mathcal{U}$. Hence, the idea that $\mathcal{U}$ is the weakest detector.

Observe that $\mathcal{SAND}$ provides the solutions using these detectors to the respective problems. Note also that the outputs of the neighborhood discovery problems that we defined $\mathcal{SNDP}$, $\mathcal{WNDP}$ and $\diamond\mathcal{NDP}$ can be used as the respective universe detectors $\mathcal{SPU}$, $\mathcal{WPU}$ and $\diamond\mathcal{PU}$. For example, if a process u in $\mathcal{SNDP}$ outputs its neighborhood set, this neighborhood set can be used to point to the real universe. Hence the following proposition.

Proposition 1 *Concrete universe detectors $c\mathcal{SPU}$, $c\mathcal{WPU}$ and $\diamond c\mathcal{PU}$ are the weakest detectors required to solve $\mathcal{SNDP}$, $\mathcal{WNDP}$ and $\diamond\mathcal{NDP}$ respectively.*

8 Detector Implementation and Future Research

Detector implementation. According to Theorem 1, the universe detectors employed by our solution to the neighborhood discovery problem are not themselves implementable in asynchronous systems. The actual implementation of the detectors can depend on the particular properties of the application. Here are a few possible ways of constructing the detectors. The nodes may be aware of the bounds on faulty nodes speed. That is, the detectors will know the maximum number of fictitious nodes they have to deal with. The nodes may contain some topological knowledge of the network. For example, the nodes may know that the network is a grid. Alternatively, the nodes may have secure communication with several trusted neighbors to ensure their presence in the selected universe.

Future research. We conclude the paper by outlining several interesting areas of research that our study suggests. Even though the concrete detectors we describe in the paper are minimal from the application perspective, it is unclear if the input that $\mathcal{SAND}$ provides is optimal. That is, is there any other information that can be gathered in the asynchronous model that can help the detector decide if a certain universe is real. We suspect that $\mathcal{SAND}$ provides the maximum possible information but we would like to rigorously prove it.

In this study, we assume completely reliable communication within a certain radius of the transmitting node R_{min}. However, in practice the propagation patterns of low-power wireless radios used in sensor and other ad hoc networks are highly irregular. See for example Zhou et al [28]. The problem of adapting a more realistic communication model is left open.

Another question is the true relationship between the universe and fault detectors. Observe that unlike fault detectors, the universe detectors require additional layout properties to enable the solution to the neighborhood discovery. It would be interesting to research if there is a complete analogue to fault detectors for this problem.

References

[1] S. Buchegger and J. Le Boudec. A robust reputation system for mobile ad-hoc. Technical Report EPFL-IC-LCA-50, EPFL, Lausanne, Switzerland, November 11 2003.

[2] S. Capkun and J.-P. Hubaux. Secure positioning in wireless networks. *IEEE Journal on Selected Areas in Communications*, 24(2):221–232, 2006.

[3] T.D. Chandra, V. Hadzilacos, and S. Toueg. The weakest failure detector for solving consensus. *Journal of the ACM*, 43(4):685–722, 1996.

[4] T.D. Chandra and S. Toueg. Unreliable failure detectors for reliable distributed systems. *Communications of the ACM*, 43(2):225–267, 1996.

[5] A. Cheng and E. Friedman. Sybilproof reputation mechanisms. In *Proceeding of the 2005 ACM SIGCOMM workshop on Economics of peer-to-peer systems P2PENCON*, pages 128–132, 2005.

[6] Sylvie Delat, Partha Sarathi Mandal, Mariusz Rokicki, and Sbastien Tixeuil. Deterministic secure positioning in wireless sensor networks. In *the ACM/IEEE International Conference on Distributed Computing in Sensor Networks (DCOSS), to appear*, June 2008.

[7] M. Demirbas and Y. Song. An RSSI-based scheme for sybil attack detection in wireless sensor networks. In *IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks WOWMOM*, pages 564–570, June 2006.

[8] J. Deng, R. Han, and S. Mishra. Security support for in-network processing in wireless sensor networks. In *ACM Workshop on Security of Ad Hoc and Sensor Networks SASN*, pages 83–93, October 2003.

[9] J. Douceur. The sybil attack. In *International Workshop on Peer-to-Peer Systems (IPTPS), LNCS*, volume 1, pages 251–260, Cambridge, MA, March 2002.

[10] J. Hwang, T. He, and Y. Kim. Detecting phantom nodes in wireless sensor networks. In *26th IEEE Conference on Computer Communications InfoCom*, pages 2391–2395, May 2007.

[11] M. Jelasity, A. Montresor, and O. Babaoglu. Towards secure epidemics: Detection and removal of malicious peers in epidemic-style protocols. Technical Report UBLCS-2003-14, University of Bologna, December 11 2003.

[12] C. Karlof and D. Wagner. Secure routing in wireless sensor networks: attacks and countermeasures. *Ad Hoc Networks*, 1(2-3):293–315, 2003.

[13] T. Kindberg and K. Zhang. Validating and securing spontaneous associations between wireless devices. In *ISW: International Workshop on Information Security*, pages 44–53, Bristol, UK, October 2003.

[14] L. Lamport, R. Shostak, and M. Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, July 1982.

[15] L. Lazos, R. Poovendran, and S. Čapkun. ROPE: Robust position estimation in wireless sensor networks. In *International Symposium on Information Processing in Sensor Networks (IPSN)*, pages 324–331, Los Angeles, CA, April 2005.

[16] L.A. Martucci, M. Kohlweiss, C. Andersson, and A. Panchenko. Self-certified sybil-free pseudonyms. In *The first ACM conference on Wireless network security WiSec*, pages 154–159, New York, NY, USA, 2008. ACM.

[17] M. Nesterenko and S. Tixeuil. Discovering network topology in the presence of byzantine faults. In *13th International Colloquium on Structural Information and Communication Complexity SIROCCO*, volume 4056 of *Lecture Notes in Computer Science*, pages 212–226, July 2006.

[18] J. Newsome, E. Shi, D. Song, and A. Perrig. The sybil attack in sensor networks: analysis and defenses. In *the third international symposium on Information processing in sensor networks IPSN*, pages 259–268, April 2004.

[19] B. Parno, A. Perrig, and V.D. Gligor. Distributed detection of node replication attacks in sensor networks. In *IEEE Symposium on Security and Privacy*, pages 49–63, May 2005.

[20] T.S. Rappaport. *Wireless communications - Principles and Practice*. Prentice Hall, 2002.

[21] N. Sastry, U. Shankar, and D. Wagner. Secure verification of location claims. In *Proceedings of the ACM workshop on Wireless security*, pages 1–10, San Diego, CA, 2003.

[22] E. Shi and A. Perrig. Designing secure sensor networks. *IEEE Wireless Communications*, 11(6), December 2004.

[23] G. Theodorakopoulos and J.S. Baras. On trust models and trust evaluation metrics for ad hoc networks. *IEEE Journal on Selected Areas in Communications*, 24(2):318–328, 2006.

[24] A. Vora and M. Nesterenko. Secure location verification using radio broadcast. *IEEE Transactions on Dependable and Secure Computing*, 3(4):369–383, 2006.

[25] A.D. Wood and J.A. Stankovic. Denial of service in sensor networks. *IEEE Computer*, 35(10):54–62, 2002.

[26] H. Yang, F. Ye, Y. Yuan, S. Lu, and W.A. Arbaugh. Toward resilient security in wireless sensor networks. In *the 6th ACM Interational Symposium on Mobile Ad Hoc Networking and Computing MobiHoc*, pages 34–45, May 2005.

[27] Q. Zhang, P. Wang, D.S. Reeves, and P. Ning. Defending against sybil attacks in sensor networks. In *Second International Workshop on Security in Distributed Computing Systems*, pages 185–191, June 2005.

[28] G. Zhou, T. He, S. Krishnamurthy, and J.A. Stankovic. Impact of radio irregularity on wireless sensor networks. In *The 2nd International Conference on Mobile Systems, Applications, and Services MobiSys*, pages 125–138, June 2004.

[29] S. Zhu, S. Setia, and S. Jajodia. LEAP - efficient security mechanisms for large-scale distributed sensor networks. In *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems SenSys*, pages 308–309, November 2003.

Unité de recherche INRIA Futurs
Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)

Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399